BB

Citation 3

(Translation of Relevant parts and Abstract)

Japanese Patent Application Laying Open (KOKAI) No. 10-11301

laid open to the public January 16, 1998

Japanese Patent Application No. 8-164867

filed June 25, 1996

Internal Priority(ies) claimed: None

Applicant(s): Masaharu IMAI, Hyogo, Japan and
YAZAKI CORPORTION, Tokyo, Japan

Inventor(s): Masaharu IMAI et al., Japanese citizens

Title of Invention: MULTITASK PROCESSOR AND MULTITASK PROCESSING CONTROL METHOD

Detailed Description of the invention:

[0031]

The microprocessor 30 includes M register files (register sets) 24-1 to 24-M, which will be described later, and a register circuit 10, which performs save/restore operations, an arithmetic circuit 26, which performs arithmetic operations according to the contents of the register files 24-1 to 24-M, an internal bus (consisting of an internal address bus and an internal data bus) 17, which connects the register files 24-1 to 24-M with the arithmetic circuit 26, which will be described later, and a bus control circuit 22, which controls the internal bus 17 and an external data bus 13, which will be described later.

[0032]

The register circuit 10 includes a save/restore internal bus 11 which is provided separately from the internal bus 17 and connected with the register files 24-1 to 24-M, which will be described later, a save/restore control circuit 12, which outputs a control signal SC for selecting a register file used for executing a task in order to control a save/restore of context according to the processing state of the arithmetic circuit 26, a save/restore bus control circuit 14, which controls the save/restore internal bus 11 as well as a save/restore data bus 15 and a save/restore address bus 16, which will be described later, and M (where M is an integer greater than or equal to 2) register files (register sets) 24-1 to 24-M each of which stores the context of one task.

[0033]

The bus control circuit 22 is connected to the external data bus 13 onto which a data memory 20, which is an external memory for storing various kinds of data, is connected. The save/restore control circuit 12 includes a register file selector circuit 12A which determines which task's context is contained in the multiple register files 24-1 to 24-M.

[0034]

The save/restore bus control circuit 14 includes a context memory 18, which is an external memory for storing saved context through the save/restore data bus 15 and the save/restore address bus 16. An overview of operation will be provided below.

[0038]

...

(ii) Control in the case where save and restore are performed

On the other hand, if any of the register files 24-1 to 24-M does not contain the context associated with the new task to be switched to, the register file selector circuit 12A finds a register file 24-Y (where Y is an integer from 1 to M) in which context to be saved into the context memory 18 according to the priorities of the tasks is stored.

[0039]

Then, the save/restore control circuit 12 uses a control signal SC to control the save/restore bus control circuit 14 to save the context stored in the register file 24-Y into the context memory 18 through the save/restore internal bus 11, and the save/restore data bus 15 and the save/restore address bus 16.

[0040]

After the context stored in the register file 24-Y is saved into the context memory 18, the save/restore control circuit 12 uses a

control signal SC again to control save/restore bus control circuit 14 to store the context to be restored to the register file 24-Y into the register file 24-Y through the save/restore data bus 15, and the save/restore address bus 16 and the save/restore internal bus 11.
[0041]

On the completion of the restore of the context to the register file 24-Y, the save/restore control circuit 12 outputs a control signal SC for activating the register file 24-Y. Then, the activated register file 24-Y becomes accessible and the arithmetic circuit 26 executes the task associated with the context according to the context stored in the register file 24-Y.
[0042]

The save/restore operations described above can be performed concurrently with the task being executed by the arithmetic circuit 26 using the internal bus 17, the bus control circuit 7, the external data bus 13, and the data memory 20 without requiring to suspend the processing of the task being executed. Accordingly, overhead of register file switching time can be reduced and consequently the performance of the entire microprocessor can be improved.
[0046]

In the initial state, as shown in FIGS. 2 and 4(a), the first task T1 is assumed to be in an execution state (= "RUN"), and the second task T2, the third task T3, and the fourth task T4 are assumed to be in an executable state (= "READY"). At this stage, the arithmetic circuit 26 performs the processing of the first task T1 based on the context of the first task T1 stored in the first register file 24-1. The result of the processing of this first task T1 is stored in the data memory 20 through the internal bus 17, the bus control circuit 22, and the external data bus 13.
[0047]

The second register file 24-2 is stored with the context of the second task T2 of the priority = 2. Further, the context memory 18 is stored with the task of the priority = 3, that is, the context of the third task T3 and the context of the fourth task T4 which are the lowest in priority at the present moment.
[0048]

When a data input waiting state and the like occur through a key board in the midst of the processing of the first task T1, as shown in FIG. 5(a), the first task T1 is put into a waiting state (= "WAIT"), and the arithmetic circuit 26 puts the second task T2 into the execution state (= "RUN") by using the context of the second task T2 stored in the second register file 24-2.

[0049]

In parallel with the execution of this second task T2, the register file selector circuit 12A of the save/return control circuit 12 makes a judgment based on the priority and state of each task shown in FIG. 5(a), and makes a decision that the context of the first task T1 stored in the first register file 24-1 which is put into an waiting state (= "WAIT") is to be saved into the context memory 18.

[0050]

Based on the judgment of this register file selector circuit 12A, as shown in the flowchart of FIG. 3, the save/restore control circuit 12 controls the save/restore bus control circuit 14 by the control signal SC, and saves the context of the first task T1 into the context memory 18 from the first register file 24-1 through the save/restore internal bus 11, the save/restore data bus 15, and the save/restore address bus 16 (step S1).

[0051]

In FIG. 6 is shown a state after the context of the first task T1 is saved into the context memory 18. In parallel with this save processing of the context of the first task T1, the arithmetic circuit 26 performs the processing of the second task T2 based on the context of the second task T2 stored in the second register file 24-2 (step S3).

[0052]

Further, the save/restore control circuit 12 restores through the save/restore internal bus 11, the save/restore data bus 15, and the save/restore address bus 16 through the control of the save/restore bus control circuit 14 by the control signal SC the context of the task which is executable (= "READY") and has the most highest priority (see FIG. 5(a)) from among the tasks where their context is stored in the context memory 18, that is, in this case, the context of the third task T3 to the first register file 24-1 (step S2).

[0053]

In FIG. 7 is shown a state after the context of the third task T3 is restored to the first register file 24-1. When a data input waiting state and the like through the key board occurs in the midst of the processing of the second task T2, as shown in FIG. 8(a), the second task T2 is put into a waiting state (= "WAIT"), and the arithmetic circuit 26 puts the third task T3 into an execution state (= "RUN") by using the context of the third task T3 stored in the first register file 24-1.

[0054]

In parallel with the execution of this third task T3, the register file selector circuit 12A of the save/restore control circuit 12 makes a decision based on the priority and state of each task shown in FIG. 8(a), and makes a decision that the context of the second task T2 stored in the second register file 24-2 in a waiting state (= "WAIT") is to be save into the context memory 18.

[0055]

Based on this decision of the register file selector circuit 12A, similarly to the processing shown in the flowchart of FIG. 3, the save/restore control circuit 12 controls the save/restore bus control circuit 14 by the control signal SC, and saves the context of the second task T2 into the context memory 18 from the second register file 24-2 through the save/restore internal bus 11, the save/restore data bus 15, and the save/restore address bus 16 (equivalent to step S1 of FIG. 3).

[0060]

According to the invention according to claim 2, in addition to the invention according to claim 1, the save/restore control means performs a decision processing of the next savable task processing and the task processing to be restored based on the priority and processing state of each task processing except for the task processing currently being executed in parallel with the task processing currently being executed, and therefore, the context corresponding to the task processing to be executed next is stored in any of the register sets, and the switching of the register sets can be promptly performed, and the switching of overhead can be reduced much further.

Explanation of the Figures:

FIG. 1

10 Register circuit
11 Save/Restore internal bus
12 Save/Restore control circuit
12A Register file selector circuit
13 Data bus
14 Save/Restore bus control circuit
15 Save/Restore data bus
16 Save/restore address bus
17 Internal bus
18 Context memory

20 Data memory

22 Bus control circuit

24 Register file

26 Arithmetic circuit

30 Micro processor

FIG. 3

#1 Start

S1 "Save/restore control circuit" saves context of task T1 into "context memory"

S2 "Save/restore control circuit" saves context of task T3 into "context memory"

S3 Task T2 uses arithmetic circuit

#2 End

Figs 4(a), 5 (a)

	Priority	Status
Task T1	1	
Task T2	2	
Task T3	3	
Task T4	4	

Citation 3

PATENT ABSTRACTS OF JAPAN

(11)Publication number : **10-011301**

(43)Date of publication of application : **16.01.1998**

(51)Int.CI. **G06F 9/46**

(21)Application number : **08-164867**

(22)Date of filing : **25.06.1996**

(71)Applicant : **IMAI MASAHARU**
YAZAKI CORP

(72)Inventor : **IMAI MASAHARU**
SHIOMI AKICHIKA
NAKANO TAKUMI
ITABASHI
MITSUYOSHI
KIROKU MASASHI
YAMASE TAKAFUMI

(54) MULTITASK PROCESSOR AND MULTITASK PROCESSING CONTROL METHOD

(57)Abstract:

PROBLEM TO BE SOLVED: To prevent the total performance of a microprocessor from decreasing even when the number of tasks to be processed is less than the number of register groups of the microprocessor.

SOLUTION: When it is necessary to save and reload a context, a saving/ reloading control means 12 saves a context stored in one of register groups 24-1 to 24-M in a context storage means 18 through saving/reloading buses 11, 15, and 16 provided independently of a bus 17 and reloads a context to be processed which is stored in the context storage means 18 to the register group, so ordinary task processing can be performed simultaneously with the saving and loading of the contexts to reduce the overhead of the saving/ reloading processing, thereby improving the total performance of the multitask processor.

Citation 3

Fig. 1

【図1】

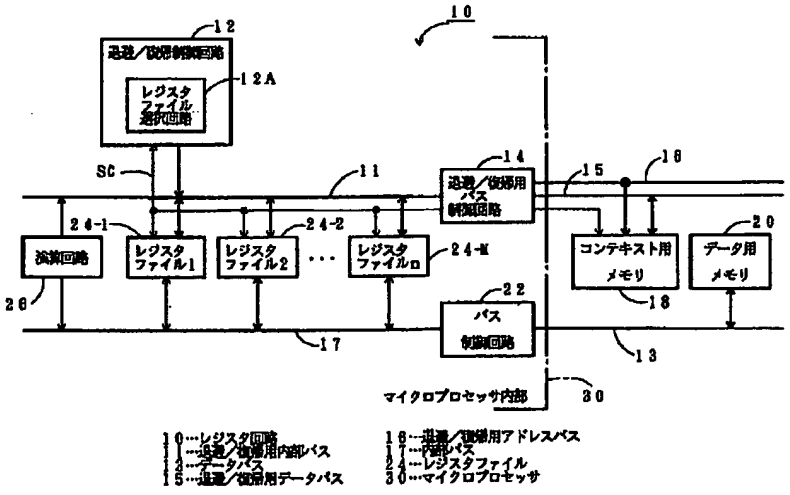


Fig. 2

【図2】

	優先度	状態
タスクT1	1	RUN
タスクT2	2	READY
タスクT3	3	READY
タスクT4	4	READY

Fig. 3

【図3】

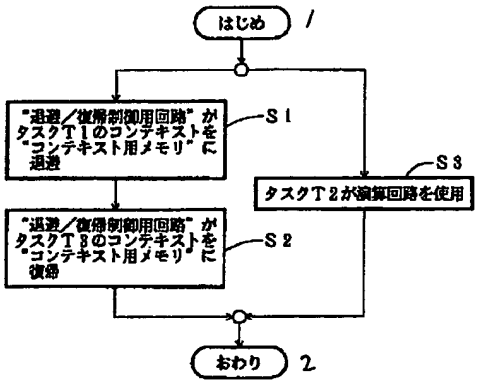


Fig. 4

【図4】

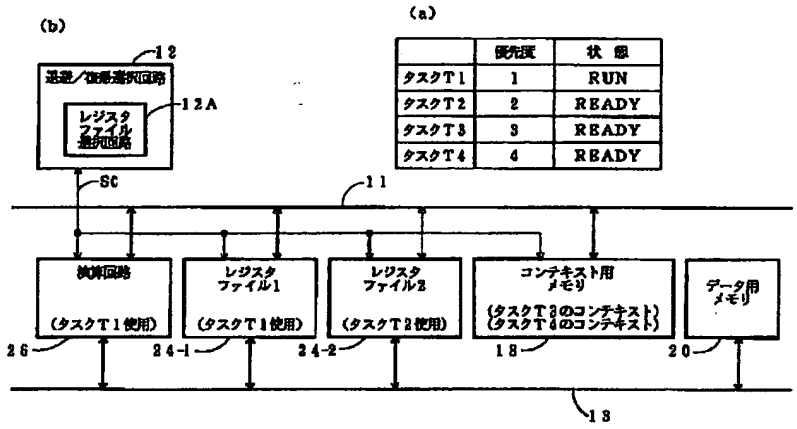


Fig. 5

【図5】

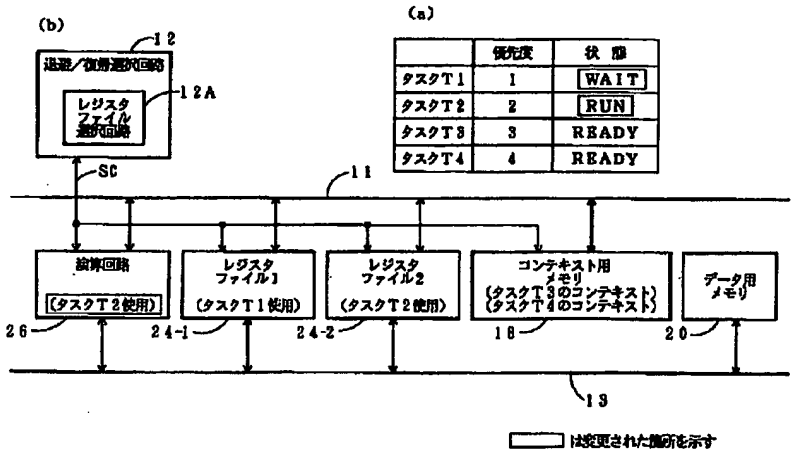


Fig. 6

【図6】

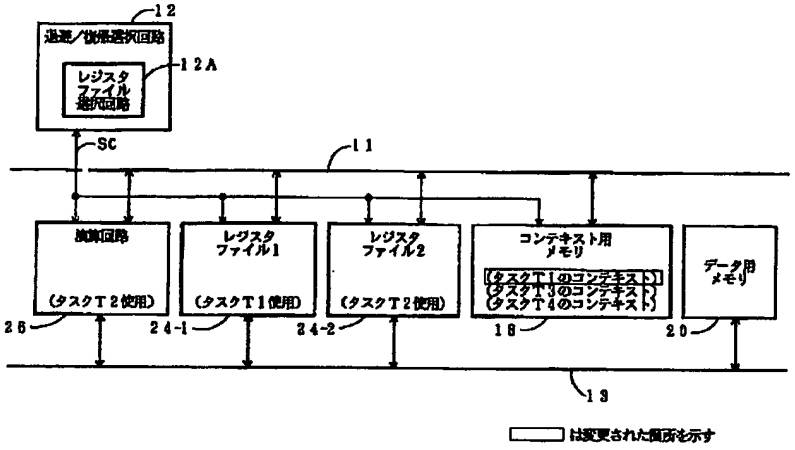


Fig. 7

【図7】

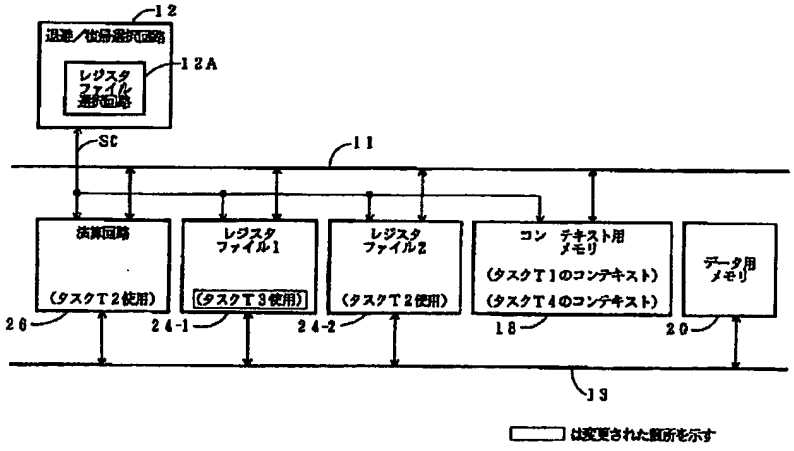


Fig. 8

【図8】

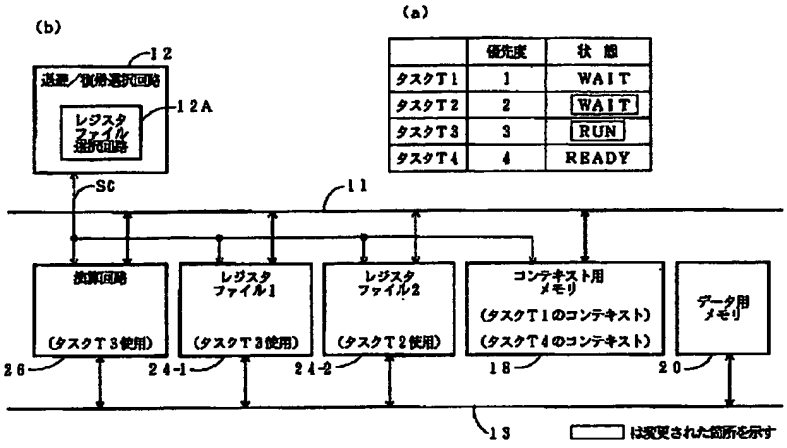


Fig. 9

【図9】

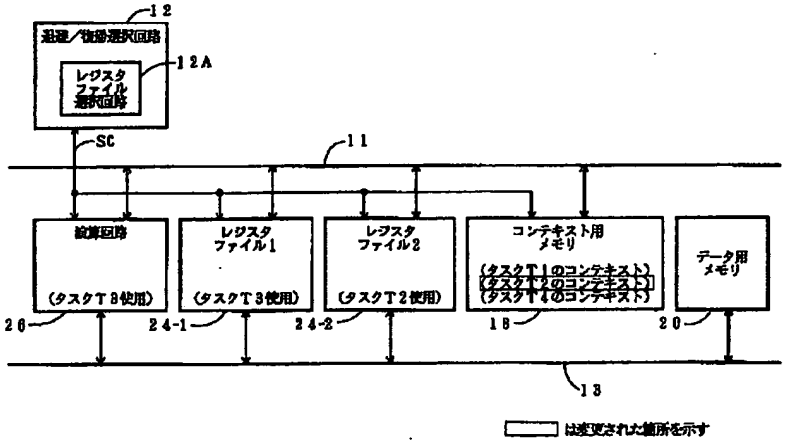


Fig. 10

【図10】

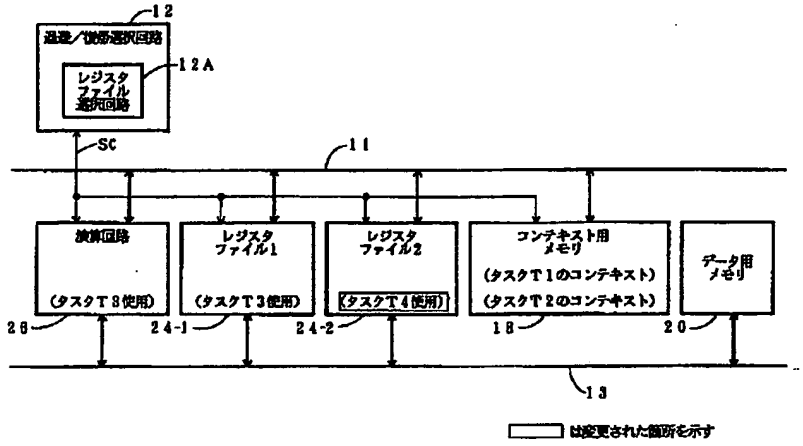
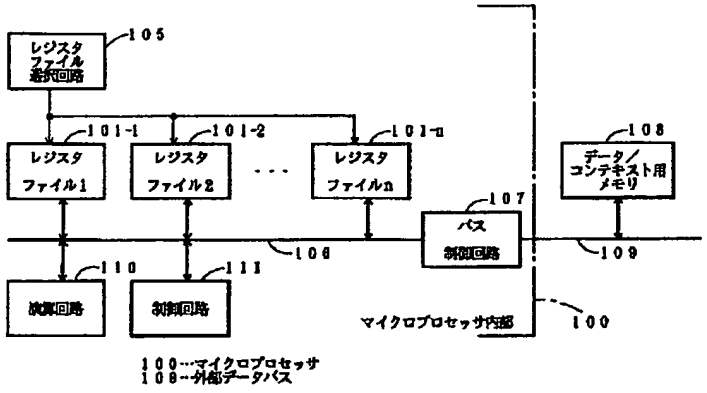


Fig. 11

【図11】



Citation 3

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（A）　(11)特許出願公開番号
特開平10－11301
(43)公開日　平成10年(1998)１月16日

(51)Int.Cl.[6]	識別記号	庁内整理番号	ＦＩ		技術表示箇所
Ｇ０６Ｆ　9/46	３１３		Ｇ０６Ｆ　9/46	３１３Ｄ	

審査請求　未請求　請求項の数５　ＯＬ　（全 12 頁）

(21)出願番号　特願平8－164867

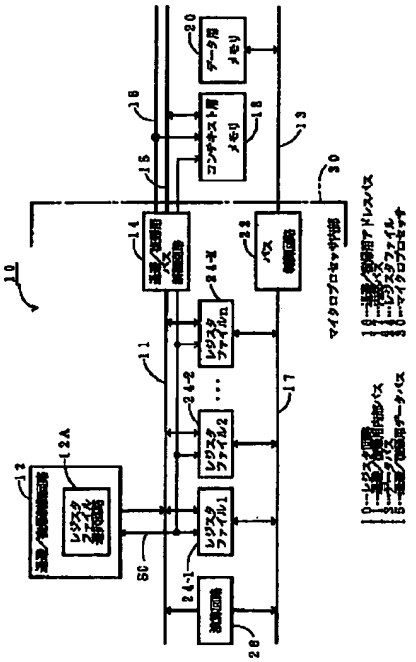
(22)出願日　平成８年(1996)６月25日

(71)出願人　592048855
今井　正治
兵庫県宝塚市雲雀丘山手２丁目15番30号
404
(71)出願人　000006895
矢崎総業株式会社
東京都港区三田１丁目４番28号
(72)発明者　今井　正治
兵庫県川西市栄町27番地　栄南団地８号棟
603号室
(74)代理人　弁理士　瀧野　秀雄　（外１名）

最終頁に続く

(54)【発明の名称】　マルチタスク処理装置及びマルチタスク処理制御方法

(57)【要約】
【課題】　処理すべきタスク数がマイクロプロセッサのレジスタ群数よりも多い場合にもマイクロプロセッサ全体のパフォーマンスを低下させない。
【解決手段】　コンテキストの退避／復帰処理を行なう必要がある場合には、退避／復帰制御手段１２は、バス１７とは別個に設けられた退避／復帰用バス１１、１５、１６を介して、いずれかのレジスタ群２４-1～２４-Mに格納されているコンテキストをコンテキスト用記憶手段１８に退避するとともに、コンテキスト用記憶手段１８に格納されている処理すべきコンテキストを当該レジスタ群に復帰させるので、通常のタスク処理をコンテキストの退避／復帰と並行して行なうことができ、退避／復帰処理のオーバーヘッドを削減して、マルチタスク処理装置１全体のパフォーマンスを向上できる。



【特許請求の範囲】
【請求項１】　各々がコンテキストを格納するＭ個（Ｍ；２以上の整数）のレジスタ群及びタスク処理を行なうべく前記Ｍ個のレジスタ群に接続されたバスを有し、選択したいずれか一のレジスタ群に格納されている前記コンテキストに基づいて、複数の前記タスク処理を順次行なうマルチタスク処理装置において、
前記コンテキストを格納するコンテキスト用記憶手段と、
前記バスとは別個に設けられ、かつ、前記Ｍ個のレジスタ群と前記コンテキスト用記憶手段との間に設けられて前記コンテキストの退避／復帰を行なうための退避／復帰用バスと、
前記退避／復帰用バスを介して、前記コンテキストを前記コンテキスト用記憶手段に退避し、あるいは、前記コンテキスト用記憶手段から前記コンテキストを前記レジスタ群に復帰させるための制御を行なう退避／復帰制御手段と、
を備えたことを特徴とするマルチタスク処理装置。
【請求項２】　請求項１記載のマルチタスク処理装置において、
前記退避／復帰制御手段は、現在実行中のタスク処理と並行して前記現在実行中のタスク処理を除く各タスク処理の優先順位及び処理状態に基づいて次に退避可能なタスク処理及び復帰すべきタスク処理の判別処理を行なうことを特徴とするマルチタスク処理装置。
【請求項３】　請求項１又は請求項２記載のマルチタスク処理装置において、
並行して処理すべき全タスク数をＬ（Ｌ；２以上の整数）とし、
Ｌ≦Ｍ
の場合には、全タスク処理に対応するコンテキストを前記Ｍ個のレジスタ群のうちのＬ個のレジスタ群に格納することを特徴とするマルチタスク処理装置。
【請求項４】　各々がコンテキストを格納するＭ個（Ｍ；２以上の整数）のレジスタ群及びタスク処理を行なうべく前記Ｍ個のレジスタ群に接続されたバスを有し、選択したいずれか一のレジスタ群に格納されている前記コンテキストに基づいて、複数の前記タスク処理を順次行なうマルチタスク処理装置のマルチタスク処理制御方法において、
現在実行中のタスク処理と並行して前記現在実行中のタスク処理を除く各タスク処理の優先順位及び処理状態に基づいて次に退避可能なタスク処理及び復帰すべきタスク処理を判別する判別工程と、
前記判別に基づいて、前記バスとは別個に設けられ、かつ、前記Ｍ個のレジスタ群と前記コンテキスト用記憶手段との間に設けられて前記コンテキストの退避／復帰を行なうための退避／復帰用バスを介して、前記退避可能なタスク処理に対応するコンテキストを前記コンテキスト用記憶手段に退避し、あるいは、前記コンテキスト用記憶手段から前記コンテキストを前記レジスタ群に復帰させるための制御を行なう退避／復帰制御工程と、
を備えたことを特徴とするマルチタスク処理制御方法。
【請求項５】　請求項４記載のマルチタスク処理制御方法において、
並行して処理すべき全タスク数をＬ（Ｌ；２以上の整数）とし、
Ｌ≦Ｍ
の場合には、前記Ｍ個のレジスタ群のうちのＬ個のレジスタ群に全タスク処理に対応するコンテキストを格納する格納工程を備えたことを特徴とするマルチタスク処理制御方法。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】本発明は、マルチタスク処理装置及びマルチタスク処理制御方法に係り、特にマルチタスキングにおいて、コンテキストスイッチング（Context switching）を行なうマルチタスク処理装置及びマルチタスク処理制御方法に関する。
【０００２】
【従来の技術】近年のマイクロプロセッサにおいては、複数のタスクを並列して処理するマルチタスキングが行なわれている。このマルチタスキングを実現する方法として、複数種類の方法があるが、このような方法の中で、最も簡単な方法としてコンテキストスイッチングが知られている。
【０００３】従来のマイクロプロセッサは、レジスタファイル（レジスタ群）を１組しか持っていなかったため、タスク切換の際には、レジスタファイルの内容であるコンテキストを外部メモリであるコンテキスト用メモリ（例えば、メインメモリ上に設ける。）に退避し、次にレジスタファイルの内容を退避したタスクの処理を行なう場合には、コンテキスト用メモリから対応するコンテキストをレジスタに復帰させていた。
【０００４】しかし、上記従来のマイクロプロセッサにおいては、タスク切換が起こる度にコンテキスト用メモリに対するアクセスが必要となる。コンテキスト用メモリ空間の切換え、例えば、メインメモリ空間の切換えはかなりの時間を要するため、マイクロプロセッサ全体としてのパフォーマンスが低下してしまうという問題点があった。
【０００５】
【発明が解決しようとする課題】上記問題点を解決すべく、複数のレジスタファイルを設け、コンテキスト用メモリ空間の切換頻度を低減することによりパフォーマンスの向上を図る方法が提案されている。
【０００６】図１１に複数のレジスタファイルを有するマイクロプロセッサの概要構成ブロック図を示す。マイクロプロセッサ１００は、それぞれ一のタスクのコンテ

キストを格納する複数のレジスタファイル１０１-1～１０１-nと、複数のレジスタファイル１０１-1～１０１-nにいずれのタスクに対応するコンテキストが格納されているかを判別し、タスクの処理に用いるレジスタファイルを選択するための制御信号を出力するレジスタファイル選択回路１０５と、レジスタファイル１０１-1～１０１-nと演算回路１１０及び制御回路１１１とを接続する内部バス（内部アドレスバス及び内部データバス）１０６と、内部バス１０６及び外部データバス１０９を制御するためのバス制御回路１０７と、を備えて構成されている。

【０００７】さらに、マイクロプロセッサ１００には、レジスタファイル１０１-1～１０１-nに格納しきれないコンテキストを退避するとともに、各種データを記憶するデータ／コンテキスト用メモリ１０８が外部データバス１０９を介して接続されている。

【０００８】次に動作を説明する。

１）　処理すべきタスク数Nがレジスタファイル数ｎ以下の場合（N≦ｎ）

この場合には、各タスクのコンテキストはいずれかのレジスタファイルに格納されていることとなる。

【０００９】従って、タスク切換えが発生した場合には、レジスタファイル選択回路１０５は、切換え先のタスクに対応するレジスタファイルを複数のレジスタファイル１０１-1～１０１-nのうちから判別し、対応するレジスタファイル１０１-X（X＝1～ｎ）のみをアクティブにするための制御信号を出力する。

【００１０】これによりそれ以降は、当該アクティブにされたレジスタファイル１０１-Xのみがアクセス可能となり、このレジスタファイル１０１-Xに格納されているコンテキストに基づいて演算回路１１０及び制御回路１１１は動作を行なうこととなる。

【００１１】従って、処理すべきタスク数Nがレジスタファイル数ｎ以下の場合には、レジスタファイル１０１-1～１０１-nに格納しきれないコンテキストが発生することがなく、各種データを記憶するデータ／コンテキスト用メモリ１０８に対して外部データバス１０９を介してアクセスする必要がないので、高速にタスク切換えを行なうことができ、マイクロプロセッサ１０１全体のパフォーマンスを向上させることができるのである。

２）　処理すべきタスク数Nがレジスタファイル数ｎより多い場合（N＞ｎ）

この場合には、N個のタスクのコンテキストのうち、ｎ個のタスクに対応するコンテキストはいずれかのレジスタファイルに格納されているとともに、（N－ｎ）個のコンテキストはデータ／コンテキスト用メモリ１０８に格納されていることとなる。

【００１２】従って、タスク切換えが発生した場合には、レジスタファイル選択回路１０５は、切換え先のタスクに対応するコンテキストがレジスタファイル１０１-1～１０１-nのいずれかに存在するか否かを判別し、いずれかのレジスタファイル１０１-1～１０１-nに当該切換え先のタスクに対応するコンテキストが存在する場合には、対応するレジスタファイル１０１-X（X＝1～ｎ）のみをアクティブにするための制御信号を出力する。

【００１３】これによりそれ以降は、当該アクティブにされたレジスタファイル１０１-Xのみがアクセス可能となり、このレジスタファイル１０１-Xに格納されているコンテキストに基づいて演算回路１１０及び制御回路１１１は動作を行なうこととなる。

【００１４】一方、いずれのレジスタファイル１０１-1～１０１-nにも当該切換え先のタスクに対応するコンテキストが存在しない場合には、レジスタファイル選択回路１０５は、タスクの優先度等に基づいてデータ／コンテキスト用メモリ１０８に退避すべきコンテキストが格納されているレジスタファイル１０１-Y（Y＝1～ｎ）を判別するとともに、バス制御回路１０７を制御することにより、当該レジスタファイル１０１-Yに格納されているコンテキストを内部バス１０６及びデータバス１０９を介してデータ／コンテキスト用メモリ１０８に退避する。

【００１５】退避が終了すると、レジスタファイル選択回路１０５は、再びバス制御回路１０７を制御することにより復帰させるべきコンテキストをデータバス１０９及び内部バス１０６を介してレジスタファイル１０１-Yに格納する。そして、レジスタファイル選択回路１０５は、コンテキストの復帰が完了すると、レジスタファイル１０１-Yのみをアクティブにするための制御信号を出力する。

【００１６】これによりそれ以降は、当該アクティブにされたレジスタファイル１０１-Yのみがアクセス可能となり、このレジスタファイル１０１-Yに格納されているコンテキストに基づいて演算回路１１０及び制御回路１１１は動作を行なうこととなる。

【００１７】以上の説明のように、処理すべきタスク数Nがレジスタファイル数ｎより多い場合には、全てのコンテキストをレジスタファイルに格納することができず、コンテキストの退避／復帰のための時間が必要となって、マイクロプロセッサ全体のパフォーマンスが低下してしまうという問題点があった。

【００１８】そこで、本発明の目的は、処理すべきタスク数がマイクロプロセッサのレジスタファイル（レジスタ群）数よりも多い場合にもマイクロプロセッサ全体のパフォーマンスを低下させることがないマルチタスク処理装置及びマルチタスク処理制御方法を提供することにある。

【００１９】

【課題を解決するための手段】請求項１記載の発明は、各々がコンテキストを格納するM個（M；２以上の整

数）のレジスタ群及びタスク処理を行なうべく前記M個のレジスタ群に接続されたバスを有し、選択したいずれか一のレジスタ群に格納されている前記コンテキストに基づいて、複数の前記タスク処理を順次行なうマルチタスク処理装置において、前記コンテキストを格納するコンテキスト用記憶手段と、前記バスとは別個に設けられ、かつ、前記M個のレジスタ群と前記コンテキスト用記憶手段との間に設けられて前記コンテキストの退避／復帰を行なうための退避／復帰用バスと、前記退避／復帰用バスを介して、前記コンテキストを前記コンテキスト用記憶手段に退避し、あるいは、前記コンテキスト用記憶手段から前記コンテキストを前記レジスタ群に復帰させるための制御を行なう退避／復帰制御手段と、を備えて構成する。

【００２０】請求項１記載の発明によれば、マルチタスク処理装置は、コンテキストの退避／復帰処理を行なう必要がない場合には、M個のレジスタ群のうち、処理すべきタスクに対応するレジスタ群に格納されているデータをバスを介してやり取りしてタスクの処理を行なう。

【００２１】そして処理に用いるレジスタ群を切換えることにより、複数のタスク処理を順次行なうマルチタスク処理を行なう。一方、コンテキストの退避／復帰処理を行なう必要がある場合、すなわち、M個のレジスタ群に処理すべきタスクに対応するコンテキストが格納されていない場合には、退避／復帰制御手段は、バスとは別個に設けられた退避／復帰用バスを介して、いずれかのレジスタ群に格納されているコンテキストをコンテキスト用記憶手段に退避するとともに、コンテキスト用記憶手段に格納されている処理すべきコンテキストを当該レジスタ群に復帰させる。

【００２２】従って、通常のタスク処理を、バスを介してコンテキストの退避／復帰と並行して行なうことができる。請求項２記載の発明は、請求項１記載の発明において、前記退避／復帰制御手段は、現在実行中のタスク処理と並行して前記現在実行中のタスク処理を除く各タスク処理の優先順位及び処理状態に基づいて次に退避可能なタスク処理及び復帰すべきタスク処理の判別処理を行なうように構成する。

【００２３】請求項２記載の発明によれば、請求項１記載の発明の作用に加えて、退避／復帰制御手段は、現在実行中のタスク処理と並行して現在実行中のタスク処理を除く各タスク処理の優先順位及び処理状態に基づいて次に退避可能なタスク処理及び復帰すべきタスク処理の判別処理を行なう。

【００２４】請求項３記載の発明は、請求項１又は請求項２記載の発明において、並行して処理すべき全タスク数をＬ（Ｌ；２以上の整数）とし、

Ｌ≦Ｍ

の場合には、全タスク処理に対応するコンテキストを前記M個のレジスタ群のうちのＬ個のレジスタ群に格納する。

【００２５】請求項３記載の発明によれば、請求項１又は請求項２記載の発明の作用に加えて、並行して処理すべき全タスク数をＬ（Ｌ；２以上の整数）とし、

Ｌ≦Ｍ

の場合には、全タスク処理に対応するコンテキストをM個のレジスタ群のうちのＬ個のレジスタ群に格納するので、タスク切換に伴うコンテキストの退避／復帰処理が生じることがなく高速に処理を行なうことができる。

【００２６】請求項４記載の発明は、各々がコンテキストを格納するM個（Ｍ；２以上の整数）のレジスタ群及びタスク処理を行なうべく前記M個のレジスタ群に接続されたバスを有し、選択したいずれか一のレジスタ群に格納されている前記コンテキストに基づいて、複数の前記タスク処理を順次行なうマルチタスク処理装置のマルチタスク処理制御方法において、現在実行中のタスク処理と並行して前記現在実行中のタスク処理を除く各タスク処理の優先順位及び処理状態に基づいて次に退避可能なタスク処理及び復帰すべきタスク処理を判別する判別工程と、前記判別に基づいて、前記バスとは別個に設けられ、かつ、前記M個のレジスタ群と前記コンテキスト用記憶手段との間に設けられて前記コンテキストの退避／復帰を行なうための退避／復帰用バスを介して、前記退避可能なタスク処理に対応するコンテキストを前記コンテキスト用記憶手段に退避し、あるいは、前記コンテキスト用記憶手段から前記コンテキストを前記レジスタ群に復帰させるための制御を行なう退避／復帰制御工程と、を備えて構成する。

【００２７】請求項４記載の発明によれば、判別工程は、現在実行中のタスク処理と並行して現在実行中のタスク処理を除く各タスク処理の優先順位及び処理状態に基づいて次に退避可能なタスク処理及び復帰すべきタスク処理を判別する。これにより退避／復帰工程は、判別工程における判別に基づいて、バスとは別個に設けられ、かつ、M個のレジスタ群と前記コンテキスト用記憶手段との間に設けられてコンテキストの退避／復帰を行なうための退避／復帰用バスを介して、前記退避可能なタスク処理に対応するコンテキストをコンテキスト用記憶手段に退避し、あるいは、コンテキスト用記憶手段からコンテキストをレジスタ群に復帰させるための制御を行なう。

【００２８】請求項５記載の発明は、請求項４記載のマルチタスク処理制御方法において、並行して処理すべき全タスク数をＬ（Ｌ；２以上の整数）とし、

Ｌ≦Ｍ

の場合には、前記M個のレジスタ群のうちのＬ個のレジスタ群に全タスク処理に対応するコンテキストを格納する格納工程を備えて構成する。

【００２９】請求項５記載の発明によれば、請求項４記載の発明の作用に加えて、格納工程は、

L≦M
の場合には、M個のレジスタ群のうちのL個のレジスタ群に全タスク処理に対応するコンテキストを格納する。
【００３０】
【発明の実施の形態】次に図面を参照して本発明の好適な実施形態を説明する。図１にマルチタスク処理装置の実施形態としてのマイクロプロセッサの概要構成ブロック図を示す。
【００３１】マイクロプロセッサ３０は、大別すると、後述のM個のレジスタファイル（レジスタ群）２４-1～２４-Mを有し、退避／復帰処理を行なうレジスタ回路１０と、レジスタファイル２４-1～２４-Mの内容に基づいて各種演算を行なう演算回路２６と、後述のレジスタファイル２４-1～２４-Mと演算回路２６を接続する内部バス（内部アドレスバス及び内部データバス）１７と、内部バス１７及び後述の外部データバス１３を制御するためのバス制御回路２２と、を備えて構成されている。
【００３２】レジスタ回路１０は、内部バス１７とは別個に設けられ、後述のレジスタファイル２４-1～２４-Mと接続されている退避／復帰用内部バス１１と、演算回路２６の処理状態に応じてコンテキストの退避／復帰を制御すべくタスクの処理に用いるレジスタファイルを選択するための制御信号SC を出力する退避／復帰制御回路１２と、後述の退避／復帰用内部バス１１並びに退避／復帰用データバス１５及び退避／復帰用アドレスバス１６を制御するための退避／復帰用バス制御回路１４と、それぞれ一のタスクのコンテキストを格納するM個（M：2以上の整数）のレジスタファイル（レジスタ群）２４-1～２４-Mと、を備えて構成されている。
【００３３】バス制御回路２２には、各種データを記憶する外部メモリとしてのデータ用メモリ２０が接続された外部データバス１３が接続されている。退避／復帰制御回路１２は、複数のレジスタファイル２４-1～２４-Mにいずれのタスクに対応するコンテキストが格納されているかを判別するレジスタファイル選択回路１２Ａを備えて構成されている。
【００３４】退避／復帰用バス制御回路１４には、退避／復帰用データバス１５及び退避／復帰用アドレスバス１６を介して、退避したコンテキストを格納するための外部メモリとしてのコンテキスト用メモリ１８を備えて構成されている。次に概要動作を説明する。
1）　処理すべきタスク数Nがレジスタファイル数M以下の場合（N≦M）
この場合には、各タスクのコンテキストはいずれかのレジスタファイル２４-1～２４-Mに格納されることとなる。
【００３５】従って、タスク切換えが発生した場合には、レジスタファイル選択回路１２Ａは、切換え先のタスクに対応するレジスタファイルを複数のレジスタファイル２４-1～２４-Mのうちから判別し、対応するレジスタファイル２４-X（X＝1～M）のみをアクティブにするための制御信号SC を出力する。
【００３６】これによりそれ以降は、当該アクティブにされたレジスタファイル２４-Xのみがアクセス可能となり、このレジスタファイル２４-Xに格納されているコンテキストに基づいて演算回路２６は動作を行なうこととなる。従って、処理すべきタスク数Nがレジスタファイル数M以下の場合には、レジスタファイル２４-1～２４-Mに格納しきれないコンテキストが発生することはなく、退避／復帰用データバス１５及び退避／復帰用アドレスバス１６を介してコンテキスト用メモリ１８をアクセスする必要がないので、高速にタスク切換えを行なうことができ、マイクロプロセッサ３全体のパフォーマンスを向上させることができるのである。
2）　処理すべきタスク数Nがレジスタファイル数Mより多い場合（N＞M）
この場合には、N個のタスクのコンテキストのうち、M個のタスクに対応するコンテキストはいずれかのレジスタファイル２４-1～２４-Mに格納されているとともに、（N－M）個のコンテキストはコンテキスト用メモリ１８に格納されていることとなる。
ｉ）　退避／復帰処理を行なわない場合の制御
そこで、タスク切換えが発生した場合には、レジスタファイル選択回路１２Ａは、内部バス１７、バス制御回路7、外部データバス１３及びデータ用メモリ２０を用いて現在行なわれている演算回路２６のタスクの処理と並行して、切換え先のタスクに対応するコンテキストがレジスタファイル２４-1～２４-Mのいずれかに存在するか否かを判別する。
【００３７】このレジスタファイル選択回路１２Ａの判別に基づいて退避／復帰制御回路１２は、いずれかのレジスタファイル２４-1～２４-Mに当該切換え先のタスクに対応するコンテキストが存在する場合には、対応するレジスタファイル２４-X（X＝1～M）のみをアクティブにするための制御信号SC を出力する。
【００３８】これによりそれ以降は、当該アクティブにされたレジスタファイル２４-Xのみがアクセス可能となり、このレジスタファイル２４-Xに格納されているコンテキストに基づいて演算回路２６はタスクの処理を行なうこととなる。
ｉｉ）　退避／復帰処理を行なう場合の制御
一方、いずれのレジスタファイル２４-1～２４-Mにも当該切換え先のタスクに対応するコンテキストが存在しない場合には、レジスタファイル選択回路１２Ａは、タスクの優先度等に基づいてコンテキスト用メモリ１８に退避すべきコンテキストが格納されているレジスタファイル２４-Y（Y＝1～M）を判別する。
【００３９】これにより退避／復帰制御回路１２は、制御信号SC により退避／復帰用バス制御回路１４を制御することにより、当該レジスタファイル２４-Yに格納さ

れているコンテキストを退避／復帰用内部バス１１並びに退避／復帰用データバス１５及び退避／復帰用アドレスバス１６を介してコンテキスト用メモリ１８に退避する。

【００４０】レジスタファイル２４-Yに格納されているコンテキストのコンテキスト用メモリ１８への退避が終了すると、退避／復帰制御回路１２は、再び制御信号ＳＣにより退避／復帰用バス制御回路１４を制御することにより、レジスタファイル２４-Yに復帰させるべきコンテキストを退避／復帰用データバス１５及び退避／復帰用アドレスバス１６並びに退避／復帰用内部バス１１を介してレジスタファイル２４-Yに格納する。

【００４１】そして、退避／復帰制御回路１２は、コンテキストのレジスタファイル２４-Yへの復帰が完了すると、レジスタファイル２４-Yをアクティブにするための制御信号SC を出力する。これによりそれ以降は、当該アクティブにされたレジスタファイル２４-Yをアクセスすることが可能となり、このレジスタファイル２４-Yに格納されているコンテキストに基づいて演算回路２６は対応するタスクの処理を行なうこととなる。

【００４２】以上の退避／復帰処理は、内部バス１７、バス制御回路7、外部データバス１３及びデータ用メモリ２０を用いて現在行なわれている演算回路２６のタスクの処理と並行して行なうことができるので、現在実行中のタスクの処理を中断する必要がなく、レジスタファイルの切換時間のオーバーヘッドを削減することが可能となる。従って、マイクロプロセッサ全体のパフォーマンスを向上させることができる。

【００４３】以上の説明のように、処理すべきタスク数Nがレジスタファイル数Mより多い場合にも、見掛け上、全てのコンテキストをレジスタファイルに格納した場合と同等の処理を行なうことができ、復帰／退避処理のために現在実行中のタスクの処理が影響を受けないように余裕を見込んだ数のレジスタファイルを用意しておけばよいので、レジスタ切換に伴うオーバーヘッドを削減するために最低限必要なレジスタファイル数は少なくてすむので、簡易な構成を達成できるにもかかわらずマイクロプロセッサ全体のパフォーマンスを向上できる。

【００４４】次に図２乃至図１０を参照してより具体的な動作を説明する。以下においては、説明の簡略化のため、レジスタファイル数M＝２の場合について説明する。この場合において、処理すべきタスクは、図２に示すように、第１タスクT1、第２タスクT2 、第３タスクT3 、第４タスクT4 の４個あり、タスクの優先度は、数字が小さいほど優先度が高く、第１タスクT1 の優先度＝１（最も優先度が高い）、第２タスクT2 の優先度＝２、第３タスクT3 の優先度＝３、第４タスクの優先度＝４（最も優先度が低い）であるとする。さらに、タスクの状態としては、図２に示すように、タスクを実行中である実行状態（図中、「ＲＵＮ」と示す。）、直ちにタスクの実行に移行できる実行可能状態（図中、「ＲＥＡＤＹ」と示す。）、キー入力待ち等の待機状態（図中、「ＷＡＩＴ」と示す。）の３状態があるものとする。また、これらのタスクの状態に関する情報（例えば、図２参照）は、退避／復帰制御回路１２が保有している。

【００４５】また、図２乃至図１０においては、図示の簡略化のため、第１タスクT1 をタスクT1 、第２タスクT2 をタスクT2 、第３タスクT3 をタスクT3 、第４タスクT4 をタスクT4 と表している。また、図４乃至図１０においては、図示の簡略化のため、バス制御回路２２及び退避／復帰用バス制御回路１４は図示を省略している。

【００４６】初期状態においては、図２及び図４（ａ）に示すように、第１タスクT1 は実行状態（＝「ＲＵＮ」）、第２タスクT2 、第３タスクT3 及び第４タスクT4は実行可能状態（＝「ＲＥＡＤＹ」）にあるものとする。この段階では、演算回路２６は、第１レジスタファイル２４-1に格納されている第１タスクT1 のコンテキストに基づいて第１タスクT1 の処理を行なっている。この第１タスクT1 の処理の結果については、内部バス１７、バス制御回路２２及び外部データバス１３を介してデータ用メモリ２０に格納される。

【００４７】そして、第２レジスタファイル２４-2には、優先度＝２の第２タスクT2 のコンテキストが格納されている。さらにコンテキスト用メモリ１８には、優先度＝３のタスク、すなわち、第３タスクT3 のコンテキスト及び現時点において優先度の最も低い第４タスクT4のコンテキストが格納されている。

【００４８】そして、第１タスクT1 の処理の途中でキーボードを介したデータ入力待ち状態等が発生すると、図５（ａ）に示すように、第１タスクT1 は待機状態（＝「ＷＡＩＴ」）となり、演算回路２６は、第２レジスタファイル２４-2に格納されている第２タスクT2 のコンテキストを用いて、第２タスクT2 を実行状態（＝「ＲＵＮ」）とする。

【００４９】この第２タスクT2 の実行と並行して、退避／復帰制御回路１２のレジスタファイル選択回路１２Ａは、図５（ａ）に示した各タスクの優先度及び状態に基づいて判別を行ない、待機状態（＝「ＷＡＩＴ」）にある第１レジスタファイル２４-1に格納されている第１タスクT1 のコンテキストをコンテキスト用メモリ１８に退避すべき旨の判別を行なう。

【００５０】このレジスタファイル選択回路１２Ａの判別に基づいて、図３のフローチャートに示すように、退避／復帰制御回路１２は、退避／復帰用バス制御回路１４を制御信号SC により制御して、第１レジスタファイル２４-1から退避／復帰用内部バス１１並びに退避／復帰用データバス１５及び退避／復帰用アドレスバス１６を介して、コンテキスト用メモリ１８に第１タスクT1

のコンテキストを退避する（ステップＳ１）。

【００５１】図6に第1タスクT1 のコンテキストをコンテキスト用メモリ１８に退避後の状態を示す。そして、この第1タスクT1 のコンテキストの退避処理と並行して演算回路２６は、第2レジスタファイル２４-2に格納されている第2タスクT2 のコンテキストに基づいて第2タスクT2 の処理を行なうこととなる（ステップＳ３）。

【００５２】さらに退避／復帰制御回路１２は、コンテキスト用メモリ１８にコンテキストが格納されているタスクのうち、実行可能状態（＝「ＲＥＡＤＹ」）にあり、かつ、最も優先度の高いタスクのコンテキスト（図５（ａ）参照）、すなわち、この場合においては第3タスクT3 のコンテキストを退避／復帰用バス制御回路１４を制御信号SC により制御し、退避／復帰用内部バス１１並びに退避／復帰用データバス１５及び退避／復帰用アドレスバス１６を介して第1レジスタファイル２４-1に復帰する（ステップＳ２）。

【００５３】図7に第3タスクT3 のコンテキストを第1レジスタファイル２４-1に復帰後の状態を示す。そして、第2タスクT2 の処理の途中でキーボードを介したデータ入力待ち状態等が発生すると、図８（ａ）に示すように、第2タスクT2 は待機状態（＝「ＷＡＩＴ」）となり、演算回路２６は、第1レジスタファイル２４-1 に格納されている第3タスクT3 のコンテキストを用いて、第3タスクT3 を実行状態（＝「ＲＵＮ」）とする。

【００５４】この第3タスクT3 の実行と並行して、退避／復帰制御回路１２のレジスタファイル選択回路１２Ａは、図８（ａ）に示した各タスクの優先度及び状態に基づいて判別を行ない、待機状態（＝「ＷＡＩＴ」）にある第2レジスタファイル２４-2 に格納されている第2タスクT2 のコンテキストをコンテキスト用メモリ１８に退避すべき旨の判別を行なう。

【００５５】このレジスタファイル選択回路１２Ａの判別に基づいて、図3のフローチャートに示した処理と同様に、退避／復帰制御回路１２は、退避／復帰用バス制御回路１４を制御信号SC により制御して、第2レジスタファイル２４-2 から退避／復帰用内部バス１１並びに退避／復帰用データバス１５及び退避／復帰用アドレスバス１６を介して、コンテキスト用メモリ１８に第2タスクT2 のコンテキストを退避する（図3のステップＳ1に相当）。

【００５６】図9に第2タスクT2 のコンテキストをコンテキスト用メモリ１８に退避後の状態を示す。そして、この第2タスクT2 のコンテキストの退避処理と並行して演算回路２６は、第1レジスタファイル２４-1に格納されている第3タスクT3 のコンテキストに基づいて第3タスクT3 の処理を行なうこととなる（図3のステップＳ３相当）。

【００５７】さらに退避／復帰制御回路１２は、図１０に示すように、コンテキスト用メモリ１８にコンテキストが格納されているタスクのうち、実行可能状態（＝「ＲＥＡＤＹ」）にあり、かつ、最も優先度の高いタスクのコンテキスト（図８（ａ）参照）、すなわち、この場合においては第4タスクT4 のコンテキストを退避／復帰用内部バス１１並びに退避／復帰用データバス１５及び退避／復帰用アドレスバス１６を介して第1レジスタファイル２４-1に復帰する（図3のステップＳ２相当）。

【００５８】そして、以下、同様にしてコンテキストの退避／復帰及びレジスタ切換えを行なってマルチタスク処理を実行する。以上の説明のように、本実施形態によれば、演算回路２６におけるタスクの実行を中断することなく、すなわち、演算回路２６におけるタスクの実行と並行してコンテキストの退避／復帰処理を行なうことができ、レジスタファイル切換に伴う退避／復帰処理のためのオーバーヘッドを削減し、マイクロプロセッサ全体のパフォーマンスを向上させることができるのである。

【００５９】

【発明の効果】請求項１記載の発明によれば、マルチタスク処理装置は、コンテキストの退避／復帰処理を行なう必要がある場合、すなわち、M個のレジスタ群に処理すべきタスクに対応するコンテキストが格納されていない場合には、退避／復帰制御手段は、バスとは別個に設けられた退避／復帰用バスを介して、いずれかのレジスタ群に格納されているコンテキストをコンテキスト用記憶手段に退避するとともに、コンテキスト用記憶手段に格納されている処理すべきコンテキストを当該レジスタ群に復帰させるので、通常のタスク処理をコンテキストの退避／復帰と並行して行なうことができ、退避／復帰処理のオーバーヘッドを削減して、マルチタスク処理装置全体のパフォーマンスを向上できる。

【００６０】請求項２記載の発明によれば、請求項１記載の発明の効果に加えて、退避／復帰制御手段は、現在実行中のタスク処理と並行して現在実行中のタスク処理を除く各タスク処理の優先順位及び処理状態に基づいて次に退避可能なタスク処理及び復帰すべきタスク処理の判別処理を行なうので、次に実行すべきタスク処理に対応するコンテキストは、いずれかのレジスタ群に格納されていることとなり、直ちにレジスタ群の切換を行なえ、切換のオーバーヘッドをより低減することが可能となる。

【００６１】請求項３記載の発明によれば、請求項１又は請求項２記載の発明の効果に加えて、並行して処理すべき全タスク数をＬ（Ｌ；２以上の整数）とし、

Ｌ≦Ｍ

の場合には、全タスク処理に対応するコンテキストをM個のレジスタ群のうちのＬ個のレジスタ群に格納するの

で、タスク切換に伴うコンテキストの退避／復帰処理が生じることがなく高速に処理を行なうことができ、マルチタスク処理装置のパフォーマンスを低下させることがない。

【0062】請求項4記載の発明によれば、判別工程は、現在実行中のタスク処理と並行して現在実行中のタスク処理を除く各タスク処理の優先順位及び処理状態に基づいて次に退避可能なタスク処理及び復帰すべきタスク処理を判別し、退避／復帰工程は、判別工程における判別に基づいて、バスとは別個に設けられてコンテキストの退避／復帰を行なうための退避／復帰用バスを介して、退避可能なタスク処理に対応するコンテキストをコンテキスト用記憶手段に退避し、あるいは、コンテキスト用記憶手段からコンテキストをレジスタ群に復帰させるための制御を行なうので、通常のタスク処理はバスを介して行ない、退避／復帰処理は並行して退避／復帰用バスを介して行なうことにより退避／復帰処理のオーバーヘッドを削減して、マルチタスク処理全体のパフォーマンスを向上できる。

【0063】請求項5記載の発明によれば、請求項4記載の発明の作用に加えて、格納工程は、

L≦M

の場合には、M個のレジスタ群のうちのL個のレジスタ群に全タスク処理に対応するコンテキストを格納するので、タスク切換に伴うコンテキストの退避／復帰処理が生じることがなく高速に処理を行なうことができ、マルチタスク処理におけるパフォーマンスを低下させることがない。

【図面の簡単な説明】

【図1】実施形態のマルチタスク処理装置の概要構成ブロック図である。

【図2】タスク処理の優先度及び状態の例を説明する図である。

【図3】実施形態の動作処理フローチャートである。

【図4】実施形態の動作説明図（その1）である。

【図5】実施形態の動作説明図（その2）である。

【図6】実施形態の動作説明図（その3）である。

【図7】実施形態の動作説明図（その4）である。

【図8】実施形態の動作説明図（その5）である。

【図9】実施形態の動作説明図（その6）である。

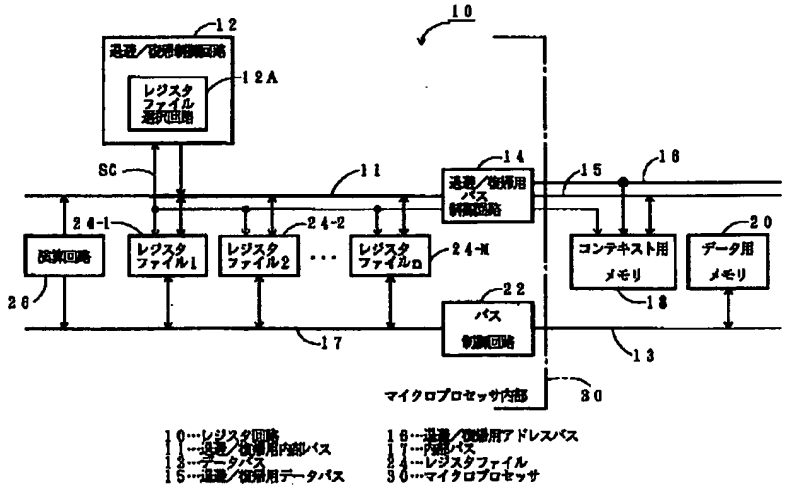
【図10】実施形態の動作説明図（その7）である。

【図11】従来のマルチタスク処理装置の概要構成ブロック図である。

【符号の説明】

10　レジスタ回路
11　退避／復帰用内部バス
12　退避／復帰制御回路
12A　レジスタファイル選択回路
13　外部データバス
14　退避／復帰用バス制御回路
15　退避／復帰用データバス
16　退避／復帰用アドレスバス
17　内部バス
18　コンテキスト用メモリ
20　データ用メモリ
22　バス制御回路
24-1～24-M　レジスタファイル（レジスタ群）
26　演算回路
30　マイクロプロセッサ

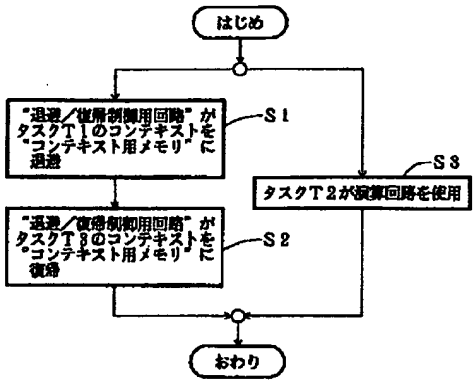
【図1】



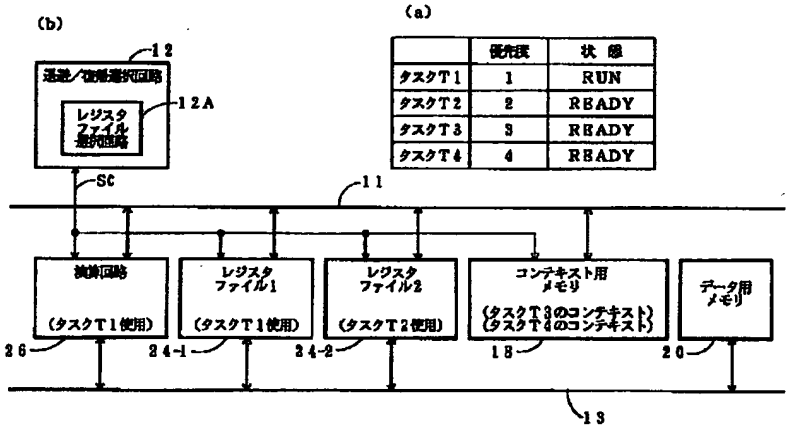
【図2】

	優先度	状態
タスクT1	1	RUN
タスクT2	2	READY
タスクT3	3	READY
タスクT4	4	READY

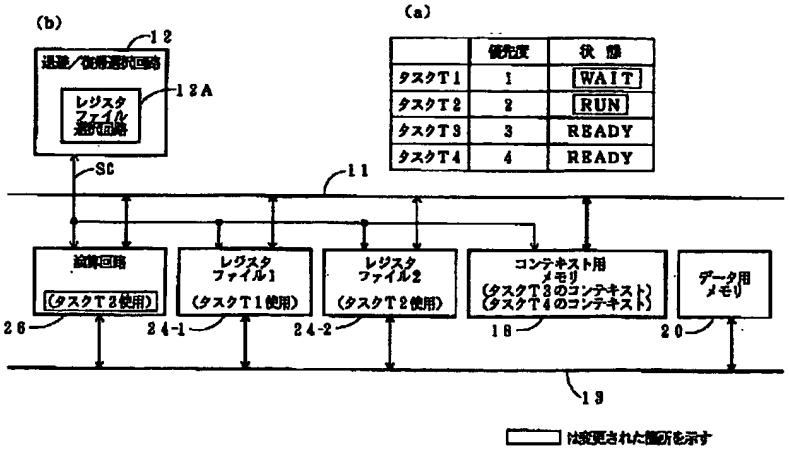
【図3】


はじめ
"退避／復帰制御用回路"がタスクT1のコンテキストを"コンテキスト用メモリ"に退避
S1
タスクT2が演算回路を使用
S3
"退避／復帰制御用回路"がタスクT3のコンテキストを"コンテキスト用メモリ"に復帰
S2
おわり

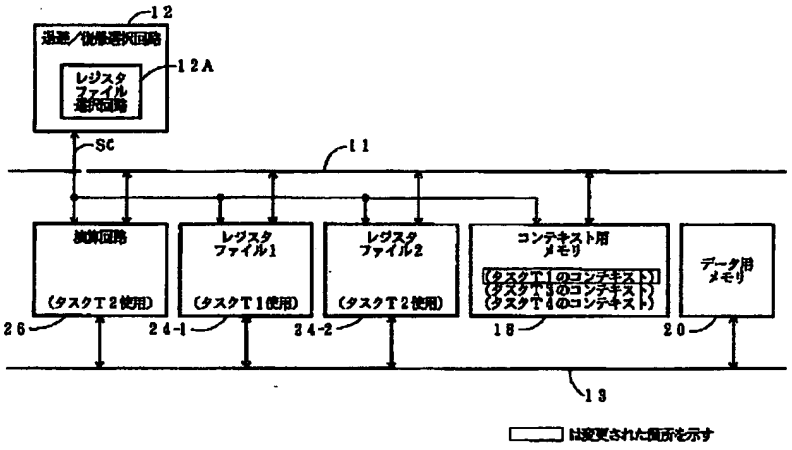
【図4】


(b)
12
退避／復帰選択回路
12A
レジスタファイル選択回路
SC
11
演算回路
(タスクT1使用)
レジスタファイル1
(タスクT1使用)
レジスタファイル2
(タスクT2使用)
コンテキスト用メモリ
(タスクT3のコンテキスト)
(タスクT4のコンテキスト)
データ用メモリ
26
24-1
24-2
18
20
13
(a)
優先度
状態
タスクT1 1 RUN
タスクT2 2 READY
タスクT3 3 READY
タスクT4 4 READY

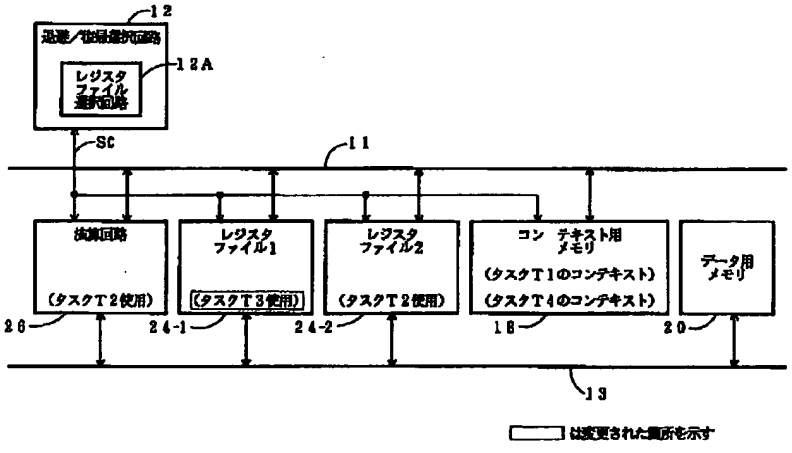
【図5】


(b)
12
退避／復帰選択回路
12A
レジスタファイル選択回路
SC
11
演算回路
(タスクT3使用)
レジスタファイル1
(タスクT1使用)
レジスタファイル2
(タスクT2使用)
コンテキスト用メモリ
(タスクT3のコンテキスト)
(タスクT4のコンテキスト)
データ用メモリ
26
24-1
24-2
18
20
13
(a)
優先度
状態
タスクT1 1 WAIT
タスクT2 2 RUN
タスクT3 3 READY
タスクT4 4 READY
は変更された箇所を示す

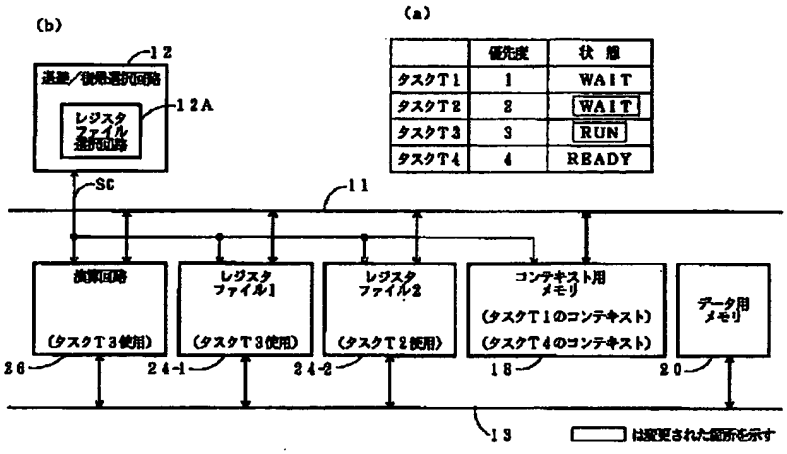
【図6】


12
退避/復帰選択回路
12A
レジスタ
ファイル
選択回路
SC
11
演算回路
(タスクT2使用)
レジスタ
ファイル1
(タスクT1使用)
レジスタ
ファイル2
(タスクT2使用)
コンテキスト用
メモリ
(タスクT1のコンテキスト)
(タスクT3のコンテキスト)
(タスクT4のコンテキスト)
データ用
メモリ
26
24-1
24-2
18
20
13
は変更された箇所を示す

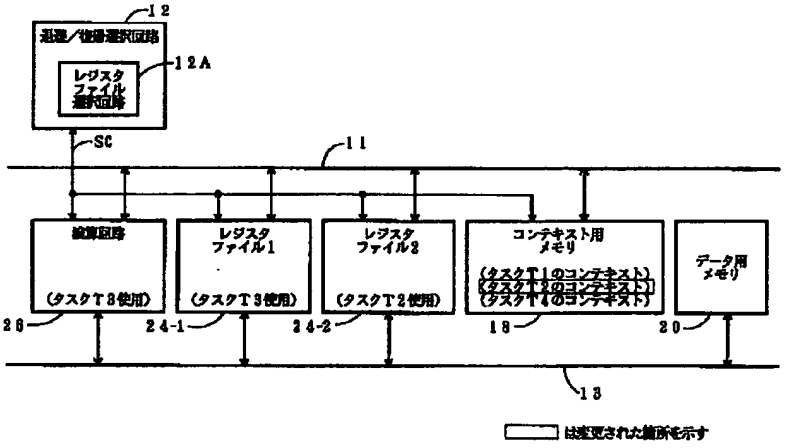
【図7】


12
退避/復帰選択回路
12A
レジスタ
ファイル
選択回路
SC
11
演算回路
(タスクT2使用)
レジスタ
ファイル1
(タスクT3使用)
レジスタ
ファイル2
(タスクT2使用)
コン テキスト用
メモリ
(タスクT1のコンテキスト)
(タスクT4のコンテキスト)
データ用
メモリ
26
24-1
24-2
18
20
13
は変更された箇所を示す

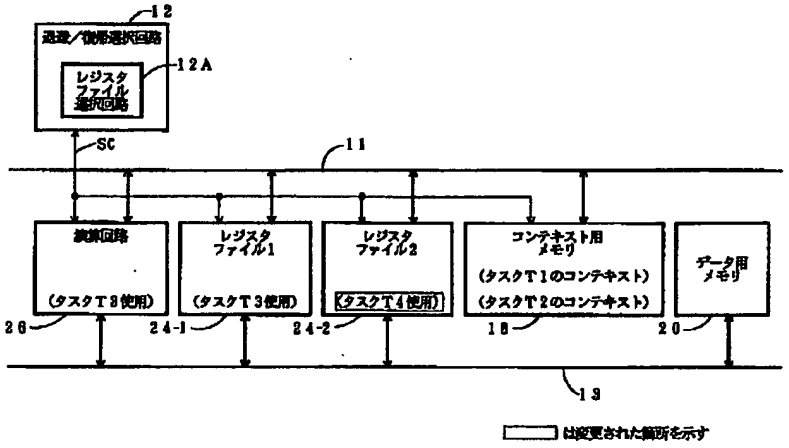
【図8】


(b)
(a)
12
退避/復帰選択回路
12A
レジスタファイル選択回路
SC
11
演算回路
(タスクT3使用)
レジスタファイル1
(タスクT3使用)
レジスタファイル2
(タスクT2使用)
コンテキスト用メモリ
(タスクT1のコンテキスト)
(タスクT4のコンテキスト)
データ用メモリ
26
24-1
24-2
18
20
13
は変更された箇所を示す
優先度
状態
タスクT1
1
WAIT
タスクT2
2
WAIT
タスクT3
3
RUN
タスクT4
4
READY

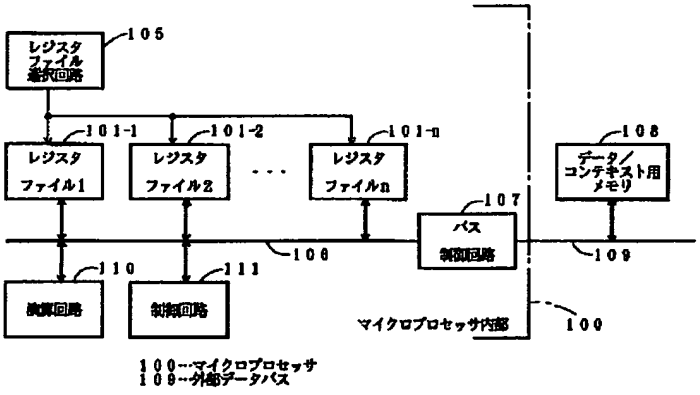
【図9】


12
退避/復帰選択回路
12A
レジスタファイル選択回路
SC
11
演算回路
(タスクT3使用)
レジスタファイル1
(タスクT3使用)
レジスタファイル2
(タスクT2使用)
コンテキスト用メモリ
(タスクT1のコンテキスト)
(タスクT2のコンテキスト)
(タスクT4のコンテキスト)
データ用メモリ
26
24-1
24-2
18
20
13
は変更された箇所を示す

【図10】



【図11】



フロントページの続き

(72)発明者　塩見　彰睦
静岡県浜松市半田町3776　医大宿舎K－544
(72)発明者　仲野　巧
愛知県宝飯郡音羽町長沢八王子49
(72)発明者　板橋　光義
静岡県裾野市御宿1500　矢崎総業株式会社内
(72)発明者　記録　真史
静岡県裾野市御宿1500　矢崎総業株式会社内
(72)発明者　山瀬　孝文
静岡県裾野市御宿1500　矢崎総業株式会社内

This Page is Inserted by IFW Indexing and Scanning Operations and is not part of the Official Record

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☒ BLACK BORDERS
- ☐ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- ☒ FADED TEXT OR DRAWING
- ☒ BLURRED OR ILLEGIBLE TEXT OR DRAWING
- ☐ SKEWED/SLANTED IMAGES
- ☐ COLOR OR BLACK AND WHITE PHOTOGRAPHS
- ☐ GRAY SCALE DOCUMENTS
- ☐ LINES OR MARKS ON ORIGINAL DOCUMENT
- ☐ REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- ☐ OTHER: _____

IMAGES ARE BEST AVAILABLE COPY.
As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.